Los Alamos
NATIONAL LABORATORY
memorandum
Computer and Computational Sciences
CCS–4:Transport Methods Group

To/MS: Distribution
From/MS: Todd J. Urbatsch/CCS–4 MS D409
Tom Evans/CCS–4 MS D409
Henry Lichtenstein, ret.
Jim Morel/CCS–4 MS D409
Phone/FAX: (505)667–3513
Symbol: CCS–4:03-08(U) (LA-UR-03-2300)
Date: April 8, 2003

Subject: Linear Error Analysis for the Residual Equilibrium Discrete Diffusion Monte Carlo (REqDDMC) Method

Executive Summary

This technical memorandum contains the details of the linear error analysis performed for the Residual Equilibrium Discrete Diffusion Monte Carlo (REqDDMC) Method. These details are meant to support the "Linear Error Analysis" section of the paper, **A Residual Monte Carlo Method for Discrete Thermal Radiative Diffusion**, by T.M. Evans, T.J. Urbatsch, H. Lichtenstein, J.E. Morel, LA-UR-02-6206, accepted by the Journal of Computational Physics in March 2003.

1. Introduction

Quantifying the gain of the REqDDMC method over the conventional EqDDMC method is not straightforward because each method converges the statistical error, σ, at a different rate. Also, the amount of work in each methods varies differently with the number of particles and cells. The usual metric for comparing linear Monte Carlo methods, the Figure-of-Merit [1] (FOM) will not suffice here, except for snapshot comparisons. The FOM,

$$\mathrm{FOM} = \frac{1}{\sigma^2 T_{\mathrm{CPU}}} , \tag{1}$$

where σ is the standard deviation and T_{CPU} is the computer (CPU) time, assumes that the methods converge the statistical error according to the Central Limit Theorem,

$$\sigma = \frac{c}{\sqrt{N}} , \tag{2}$$

where N is the number of particles. The FOM is used typically to measure the effectiveness of variance reduction techniques in minimizing the constant c in Eq. (2).

2. Gain

To compare REqDDMC with EqDDMC, we will compare errors as a function of computer time for the linear case of a single timestep. For each method, we have analytical expressions for both the convergence rate and the computer time. The coefficients of these expressions are unknown, but we can determine them heuristically. Once the coefficients are determined, we can parameterize in numbers of particles, particles per stage, stages, and residual convergence criterion to functionally determine the residual gain,

$$\mathrm{Gain(t)} = \frac{\epsilon_{\mathrm{EqDDMC}}(t)}{\epsilon_{\mathrm{REqDDMC}}(t)} \,, \tag{3}$$

as a function of computer time, where the error, ϵ, for each method is evaluated for the same computer time. Then we can plot the gain as a function of computer time.

Our linear error analysis shows rapidly increasing residual gains of up to 10 orders of magnitude. This residual gain represents what can be achieved in a linear Monte Carlo calculation or in one linearized timestep of a nonlinear Monte Carlo calculation. The actual gain in a REqDDMC calculation depends on the linear, residual gain each timestep and the propagation of deterministic and stochastic errors over multiple timesteps.

3. **Computer Time**

The average computer time necessary to track a particle is the same for both the EqDDMC and REqDDMC methods. That time will depend on both the timestep and the optical thickness of the cells. The cell-dependent source and tally calculations before and after a REqDDMC stage are essentially the same as those before and after an EqDDMC timestep, so we can generalize the cost of that work with an average time per cell per stage, understanding that an EqDDMC timestep has only one stage. The CPU time for an EqDDMC timestep is

$$t_{\mathrm{EqDDMC}} = t_{\mathrm{p}} N_{\mathrm{EqDDMC}} + t_{\mathrm{cs}} N_{\mathrm{cells}} \,, \tag{4}$$

where N_{cells} is the number of cells, N_{EqDDMC} is the number particles per EqDDMC timestep, t_{p} is the average CPU time per particle, and t_{cs} is the CPU time per cell per stage. Both t_{p} and t_{cs} are the same for the EqDDMC and REqDDMC methods. The CPU time for an REqDDMC timestep is

$$t_{\mathrm{REqDDMC}} = (t_{\mathrm{p}} N_{\mathrm{ps}} + t_{\mathrm{cs}} N_{\mathrm{cells}}) N_{\mathrm{stages}} \,, \tag{5}$$

where N_{ps} is the number of particles per stage and N_{stages} is the number of stages. For a single timestep, the total number of REqDDMC particles is

$$N_{\mathrm{REqDDMC}} = N_{\mathrm{ps}} N_{\mathrm{stages}} \,. \tag{6}$$

All calculations were performed on a LINUX-based Pentium IV processor.

4. **Errors**

Equation (2), $\sigma_{\mathrm{EqDDMC}} = c/\sqrt{N}$, is the expression for the statistical error convergence in one timestep of EqDDMC. The expression for the exponential error convergence in one timestep of the REqDDMC method is

$$\sigma_{\mathrm{REqDDMC}} = e^{-bN} \,, \tag{7}$$

where $b > 0$ [2].

We may estimate the errors by either running replicate non-analytic calculations and performing ensemble statistics or by comparing individual calculations to analytic solutions. Then, by changing N and determining a new value of the error, we can back out the constants b and c.

5. **Results and Conclusions**

For the linear error analysis, we consider one timestep of a steady-state, homogeneous, infinite medium problem, which has an analytic solution equal to its initial condition. In our simulation, the nominal cell size is 0.005 cm, the nominal time step is 10^{-4} shakes, the density is 3.0 g/cc, the absorption coefficient is a constant 100.0 cm^2/g, and the specific heat is 0.1 Jerks/g/keV. The problem has 100 cells with reflecting boundary conditions and an initial temperature of 1 keV.

We determine the CPU time per particle, t_{p}, by running a set of one-cycle problems with a number of particles ranging from 1 to 10^6. We found t_{p} to be relatively insensitive to the cell size and timestep in the neighborhood of our nominal parameters, as shown in Table 1. Given the data in

TABLE 1: CPU time per particle, t_{p}, in μsec.

	Δt (sh)					
Δx (cm)	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
0.5	57	57	57	57	57	57
0.05	49	48	47	48	48	48
0.005	57	48	47	48	47	47
0.0005	153	48	48	47	47	47

Table 1, we conservatively set the CPU time per particle as

$$t_{\mathrm{p}} = 5 \times 10^{-5}\,\mathrm{sec}\,. \tag{8}$$

To determine the CPU time per cell per stage, t_{cs}, we ran a set of 1000–cycle EqDDMC problems with a small number, 10, of particles, and with the number of cells varying up to 10^5. Given the resulting runtimes in Table 2, we set the CPU time per cell per stage as

TABLE 2: Runtime data from 1000–cycle runs for increasing numbers of cells.

N_{cells}	runtime (sec)
1	0.595
10	0.737
100	0.507
1000	5.048
10000	52.933
100000	537.373

$$t_{\mathrm{cs}} = 5.4 \times 10^{-6}\,\mathrm{sec}\,. \tag{9}$$

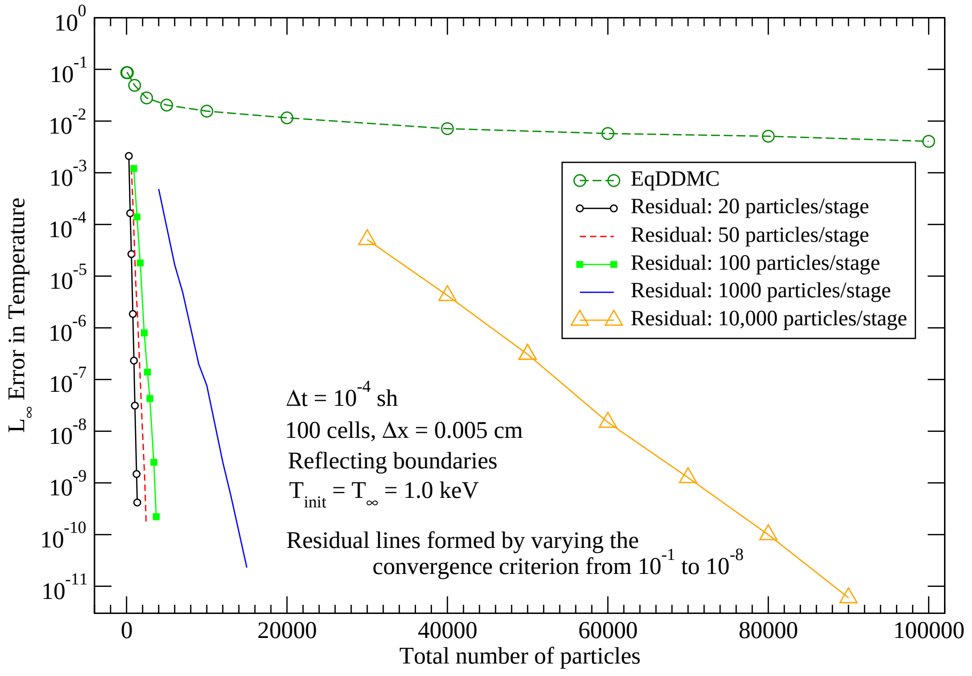

FIG. 1: The exponential error convergence of the REqDDMC method is shown compared to the inverse-root-N convergence of the stock EqDDMC for the linear case of one timestep.

The errors for one cycle of both EqDDMC and REqDDMC are shown in Fig. 1. The errors are measured as the l_∞ norm of the temperature with respect to the analytic solution of unity. The REqDDMC curve for each selection of $N_{\rm ps}$ was formed by running REqDDMC eight times, varying the convergence criterion from 10^{-1} to 10^{-8} in powers of ten. This particular problem required at least 20 particles per stage. Requesting fewer than 20 particles in these problems resulted in no particles being apportioned because of the stratified sampling we use for the source and the uniformity of the source across the entire problem. From the error data, we ascertain the error coefficients. For EqDDMC,

$$c = 1.4\ . \tag{10}$$

For the residual method, b is a function of the number of particles per stage and is determined heuristically by the ratio of two points from each residual curve in Fig. 1. Table 3 shows the values of b.

The linear error analysis requires knowledge of the number of stages required for the timestep. Figure 2 shows the number of stages required for each set of REqDDMC calculations. The curves are described approximately by

$$N_{\rm stages} \approx k \log_{10}(\frac{\epsilon}{10})\ , \tag{11}$$

where k is a function of $N_{\rm ps}$ and is tabulated in Table 3.

Now we have enough information to determine the residual gain as a function of CPU time. We set

$$t_{\rm EqDDMC} = t_{\rm REqDDMC}\ , \tag{12}$$

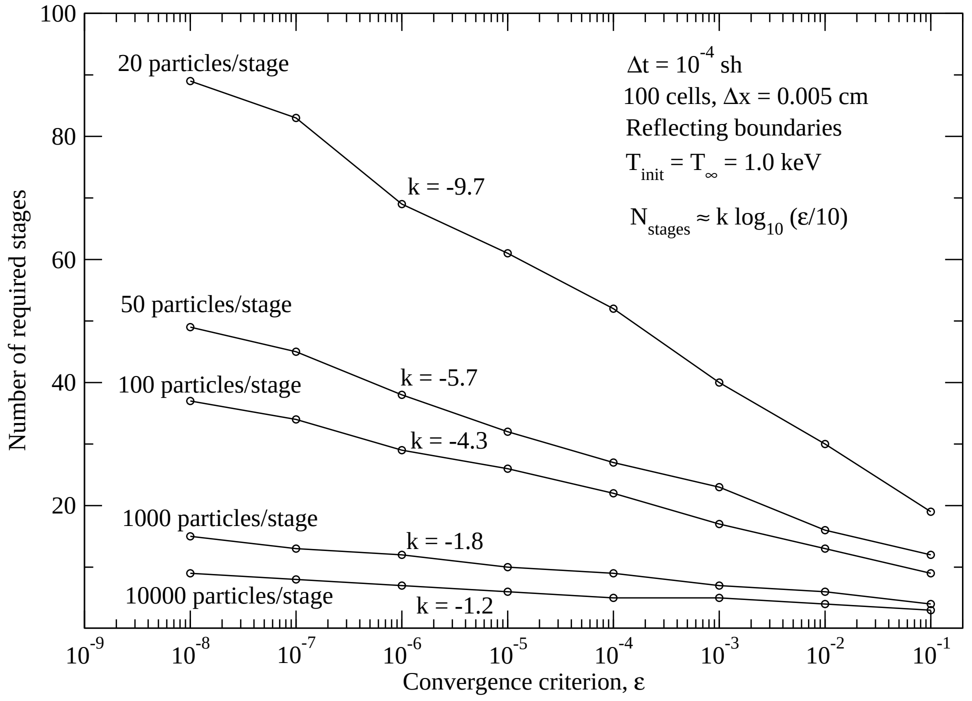

FIG. 2: Required number of stages.

from Eqs. (4) and (5), and solve for $N_{\rm EqDDMC}$ to obtain

$$N_{\rm EqDDMC} = N_{\rm ps} N_{\rm stages} + \frac{t_{\rm cs}}{t_{\rm p}} N_{\rm cells} (N_{\rm stages} - 1) \ . \tag{13}$$

The analysis considered two cases of numbers of cells, $N_{\rm cells} = 100$ and $N_{\rm cells} = 10^6$. For each $N_{\rm cells}$, we looped over $N_{\rm ps}$ values of 20, 50, 100, 1000, 10000. For each $N_{\rm ps}$, we obtain k and b from Table 3. Then we loop over the convergence criteria from 10^{-1} to 10^{-8} in powers of ten. For each case, we calculate $N_{\rm stages}$ from Eq. (11), $N_{\rm EqDDMC}$ from Eq. (13), $N_{\rm REqDDMC}$ from Eq. (6), the CPU time from both Eq. (4) and Eq. (5) (and checking that they are the same), and, finally, the residual gain as the ratio of the EqDDMC error to the REqDDMC error from Eqs. (2) and (7), respectively. The residual gains, which approach 10 orders of magnitude, are shown in Fig. (3).

TABLE 3: Values of the coefficient, b, in Eq. (7), the expression for the residual EqDDMC error convergence, and the coefficient, k, in Eq. (11), the expression for the number of stages.

$N_{\rm ps}$	b	k
20	1.47e-2	-9.7
50	8.58e-3	-5.7
100	5.54e-3	-4.3
1000	1.53e-3	-1.8
10000	2.66e-4	-1.2

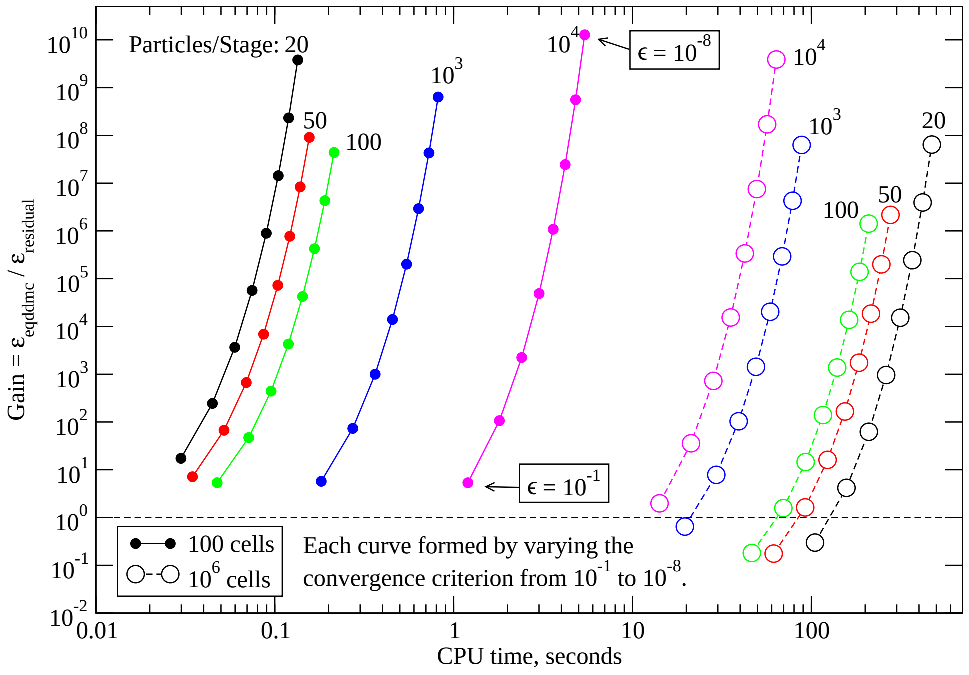

FIG. 3: Residual gain as a function of CPU time for 100 cells and 10^6 cells and varying numbers of particles per stage.

Observing the gain curves in the 100-cell problem, we see the effects of the competing functions of the overall exponential convergence and the first stage's inefficient inverse-root-N convergence. The 10^4 particles in just one stage takes longer than the 20 particles/stage case for any convergence criterion. The deterministic overhead work for each stage is insignificant.

Observing the gain curves in the 10^6-cell problem, we see that the deterministic overhead work for each stage dominates the cost of transporting particles. In fact, with too few particles per stage, more stages are required, and the overall residual gain diminishes. Note that the dependence on number of particles per stage is reversed: more particles per stage is better than less for the larger number of cells. The 10^6-cell gain curves were generated with the 100-cell parameters, therefore they do not represent the fact that this problem actually requires 10^6 requested particles (twice that on the first stage) to sample the phase space sufficiently. Nonetheless, these curves accurately represent the time requirements for transporting particles and for performing the deterministic overhead for each stage.

The balance between transport work, deterministic overhead work per stage, stability, and adequate sampling will become more important when we extend the REqDDMC method to two and three spatial dimensions.

References

[1] J. F. Briesmeister, MCNP-A General Monte Carlo N-Particle Transport Code, no. LA-12625-M, Version 4B, Los Alamos National Laboratory, 1997.

[2] J. Halton, Sequential monte carlo techniques for the solution of linear systems, Journal of Scientific Computing 9 (2) (1994) 213–257.

TJU:tju

Distribution:
Gordon Olson, CCS–4, MS D409
Tom Evans, CCS–4, MS D409
Jim Morel, CCS–4, MS D409
Todd Urbatsch, CCS–4, MS D409
CCS-4 Files